MFRF Manual

Efficient and accurate phenotype imputation in millions of individuals for increasing GWAS power

Version: 0.1.0

Date: November, 20th 2022

Author: Linlin Gu and Ming Fang

Maintainer:

Linlin Gu: linlin-gu@outlook.com

Prof. Ming Fang: fangming618@126.com

Table of Contents

1 Brief introduction

Genetic association studies have yielded a wealth of biological discoveries. What’s more, the large data sets represent an important way to move beyond simple genome wide association studies (GWAS) with great potential. However, modern data acquisition based on high-throughput technology is often facing the problem of missing data. Missing value imputation offers a solution to this problem. Here we address the central issue of missing phenotypes in genetic association studies (such as UK Biobank dataset). We herein propose a new method which can accomplish non-parametric missing value imputation for genetic studies and use a mixed fast random forest algorithm to fit the model, named MFRF. What’s more, MFRF does not need genotype dataset, and thus is computational fast. MFRF exhibits attractive computational efficiency and can cope with large data.

2 MFRF function

MFRF	Imputing missing phenotypes by using the mixed fast random forest algorithm.

2.1 Description

Imputing missing phenotypes by using the mixed fast random forest method.

2.2 Usage

MFRF(xmis, Total_maxiter = 20, maxiter = 20, num.trees = 100, mtry = floor(sqrt(ncol(xmis))), initialLinearEffects = 0, ErrorTolerance = 0.001, targetID = NULL, missing_size = 500, seed = NULL, replace = TRUE, decreasing = TRUE, verbose = TRUE, sampsize = NULL, max.depth = NULL, xtrue = NA)

2.3 Arguments

xmis	A vector, matrix or data frame with missing values.
Total_maxiter	Stop after how many iterations. (default = 10)
maxiter	The maximum iteration times of mixed fast random forest. (default = 10)
num.trees	How many trees are grown in the mixed fast random forest

(default = 100).

mtry How many variables should be tried randomly at each node.

initialLinearEffects The initial values for linear effects. (default = 0)

ErrorTolerance The tolerance for log-likelihood. (default = 0.001)

targetID The columns of the target trait, and used only for cross-validation operation (default = NULL)

missing_size The missing values size of settings in target trait, and used only for cross-validation operation (default = 500)

seed Random seed. Default is NULL, which generates the seed from R. Set to 0 to ignore the R seed.

replace (boolean) If TRUE bootstrap sampling (with replacements) is performed, else subsampling (without replacements)

decreasing (boolean) If TRUE the columns are sorted with decreasing amount of missing values

verbose (boolean) If TRUE then missForest returns error estimates, runtime and if available true error during iterations.

sampsize List of size(s) of sample to draw.

max.depth Maximal tree depth. A value of NULL or 0 (the default) corresponds to unlimited depth, 1 to tree stumps (1 split per tree).

xtrue The complete data (a vector, matrix or data frame).

2.4 Value

Return a list, the list contains:

ximp The imputed data (a vector, matrix or data frame).

score Score of the imputation for the target trait.

value Oberved phenotypic values and imputed phenotypic values of MFRF (a data frame).

bias Bias after linear fitting of oberved phenotypic values and imputed phenotypic values of MFRF.

3 MFRF.Eval function

MFRF.Eval MFRF evaluation indicator.

3.1 Description

The correlation coefficient between these imputed phenotypes and their true hidden values.

3.2 Usage

MFRF.Eval (ximp, xmis, xtrue)

3.3 Arguments

ximp	The imputed data (a vector, matrix or data frame).
xmis	The data with missing values.
xtrue	The complete data.

3.4 Value

Return the correlation coefficient between the real values and the imputed values.

4 prodNA function

prodNA Produce missing values.

4.1 Description

This R script contains the function to produce missing values in a given and data set completely at random.

4.2 Usage

prodNA(x, noNA, seed)

4.3 Arguments

x	A vector, matrix or data frame.
noNA	Proportion of missing values to add to x. In case x is a data frame, noNA can also be a vector of probabilities per column or a named vector (see examples).
seed	An integer seed.

5 sim_G function

sim_G	Simulated Genome Relationship Matrix.

5.1 Description

The function simulates the construction of a genome relational matrix.

5.2 Usage

sim_G(N, k, fam_size)

5.3 Arguments

N	The number of individuals and must be a positive integer.
k	Coefficient of kinship and the value ranges from 0 to 1.
fam_size	The size of the family, fam_size must be a positive integer and must divide N.

6 sim_pheno function

sim_pheno	Simulated phenotype.

6.1 Description

This function simulates the phenotypes for individuals.

6.2 Usage

sim_pheno(N=N, P=P, K=G, h2=rep(0.6, P), B, E)

6.3 Arguments

N	The number of individuals.
P	The number of phenotypes.
K	A genome relational matrix.
h2	The heritability of each phenotype in individuals.
B	Genetic covariance. (allow the missing)

E   Environmental or residual covariance. (allow the missing)

7 imputeUnivariate function

imputeUnivariate   Univariate Imputation.

7.1 Description

Fills missing values of a vector, matrix or data frame by sampling with replacement from the non-missing values. For data frames, this sampling is done within column.

7.2 Usage

imputeUnivariate (xmis, v = NULL, seed = NULL)

7.3 Arguments

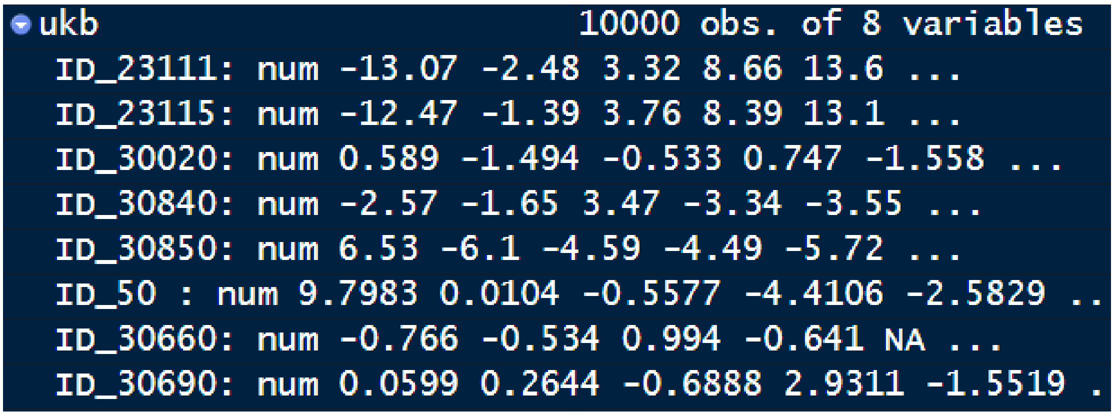
xmis   A vector, matrix or data frame with missing values.

v   A character vector of column names to impute (only relevant if x is a data frame). The default NULL imputes all columns.

seed   An integer seed.

8 Build in data

An example dataset ‘ukb’ is the UK Biobank datasets. The dataset including a data frame (10000*8), each row represents 8 phenotypes information for individuals. The ‘ukb’ can be loaded with data(ukb).



```
ukb                           10000 obs. of 8 variables
  ID_23111: num -13.07 -2.48 3.32 8.66 13.6 ...
  ID_23115: num -12.47 -1.39 3.76 8.39 13.1 ...
  ID_30020: num 0.589 -1.494 -0.533 0.747 -1.558 ...
  ID_30840: num -2.57 -1.65 3.47 -3.34 -3.55 ...
  ID_30850: num 6.53 -6.1 -4.59 -4.49 -5.72 ...
  ID_50 : num 9.7983 0.0104 -0.5577 -4.4106 -2.5829 ..
  ID_30660: num -0.766 -0.534 0.994 -0.641 NA ...
  ID_30690: num 0.0599 0.2644 -0.6888 2.9311 -1.5519 .
```

8.1 Running build-in data

```
library("MFRF")
data(ukb)
ukb.mis <- ukb
MFRF.imp <- MFRF(ukb.mis, Total_maxiter = 20, maxiter = 20, num.trees = 100,
                 mtry = floor(sqrt(ncol(ukb.mis))), initialLinearEffects = 0,
                 ErrorTolerance = 0.001, targetID = 7, missing_size = 500, seed
                 = 123, replace = TRUE, decreasing = TRUE, verbose = TRUE,
                 sampsize = NULL, max.depth = NULL, xtrue = NA)
```

9 Code availability

The source code of MFRF is freely available.

10 How to access help

If users have any bugs or issues or any suggestions are available, feel free to contact:

Linlin Gu: linlin-gu@outlook.com

Prof. Ming Fang: fangming618@126.com